

IPC-2503

Manual of Style and Usage for IPC-25XX Standards

Working draft

IPC-2503
April 2004

2215 Sanders Road, Northbrook, IL 60062-6135
Tel. 847.509.9700 Fax 847.509.9798
www.ipc.org

Version history:

Date	**Version**	**Authors**	**Description**
3/3/2004	0.1	Niko	Initial draft
4/5/2004	0.2	Dieter	CAMX conference call input
4/29/2004	0.3	Dieter	CAMX conference call input

TABLE OF CONTENTS

Style Sheet of Naming Conventions Used in IPC-25xx Series

1 SCOPE

This standard establishes the style and conventions to be used in the development of the IPC-25XX standards that define XML Schema requirements for sending data over private network or the internet. The details combine the characteristics of the W3C recommendations regarding XML and those that have been established through the consensus process of the committee members developing the 25XX Web based standards.

Descriptions provided herein include schema structures, graphic representation, and table layout formats to be used in the explanation of the standard requirements. The text details shall use these concepts wherever possible in order to maintain a consistency between and among the various standards identified in the IPC-25xx series.

1.1 Purpose

The purpose of this standard is to ensure the potential of harmonizing element and attribute descriptions and thus minimize ambiguity and duplication, with different characteristics describing the same objects, in the IPC-25xx standards. It should be noted that although the requirements are standardized in many instances the domain being covered by the standard might take precedence. However, every attempt should be made to incorporate the principles set forth in this standard into any of the IPC-25xx standards.

2 Applicable Documents

The following documents contain provisions that when referenced become a mandatory part of this standard to the extent specified herein.

IPC-T-50 *Terms and Definitions for Printed Boards and Printed Board Assemblies*

IPC-2500 *Definition for Web-Based Exchange of XML Data Series [Message Broker]*

IPC-2510 *Product Manufacturing Description Data and Transfer methodology Series [GenCAM]*

IPC-2530 *Standard Recipe File Format Series [SRFF]*

IPC-2540 *Electronic Manufacturing Shop Floor Equipment Communication Series [CAMX]*

IPC-2550 *Manufacturing Execution Systems Communication Series [MES]*

IPC-2570 *Electronics Manufacturing Supply Chain Communication Product Data eXchange [PDX]*

IPC-2580 *Printed Board Assembly Products Manufacturing Description Data Transfer Methodology*

3 General Requirements

The following general requirements apply to the style of documents in the IPC-25XX standards

3.1 Terms and Definitions

The definition of terms shall be in accordance with IPC-T-50 and the following:

Cardinality The variation of the occurrences of a particular attribute or element

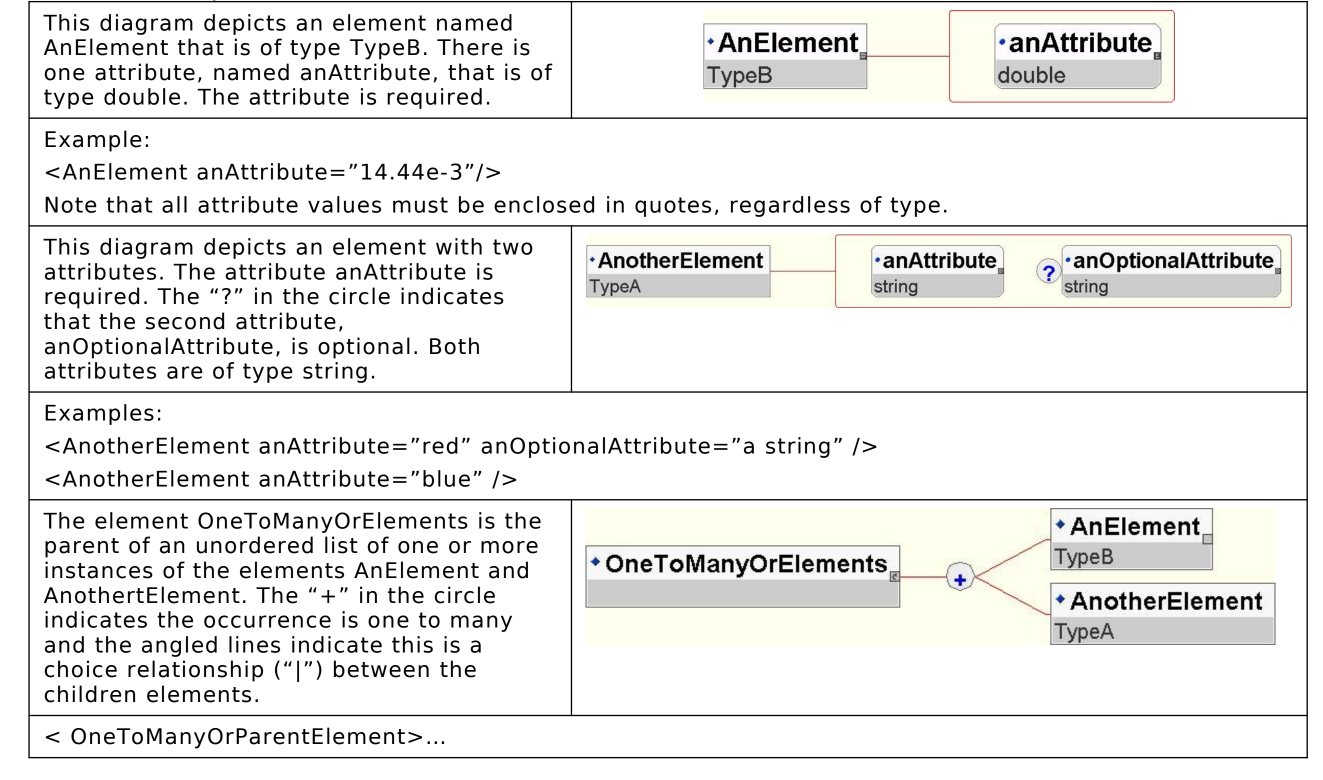
3.2 Documentation Convention

The XML file format standard and the XML Schema definition language standard, as defined the by World Wide Web Consortium (W3C), have been adopted by IPC for use in the IPC-25XX series of standards.

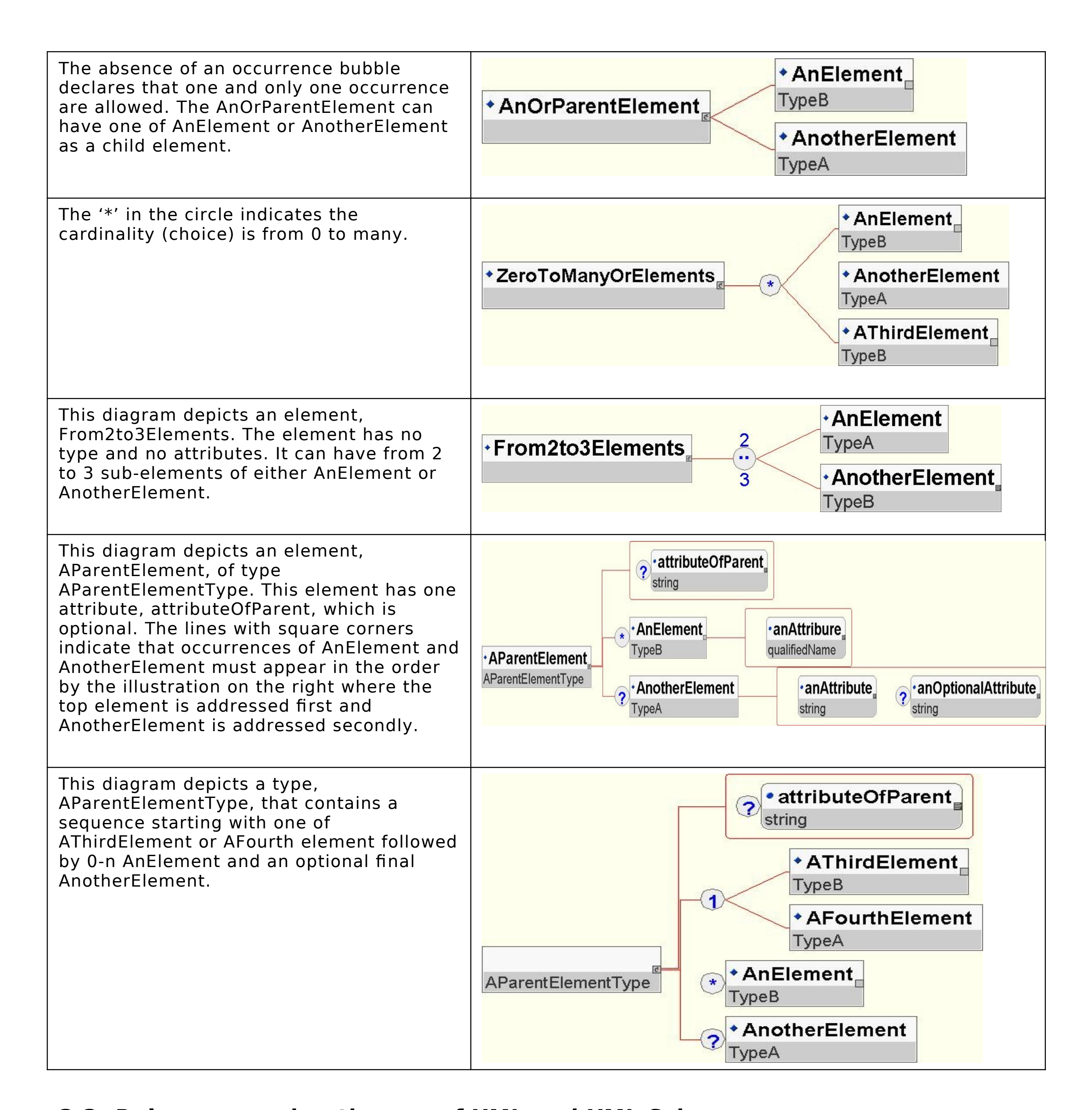
In addition to the text based schema notation this document provides graphical representation of the structure of the file format. The XML diagrams are designed to effectively illustrate the structure and cardinality of elements and attributes that make up any IPC-25XX file. The notation in the graphics does not provide a complete visualization of the schema definition for the file format, but it does provide a good top down overview. Should there be any conflict between the graphical notation and the schema notation, the authoritative definition is the schema notation.

Table 1 provides an overview of the graphical notation used in the IPC-25XX Standards.

Table 1 Graphical Notation Overview



This diagram depicts an element named AnElement that is of type TypeB. There is one attribute, named anAttribute, that is of type double. The attribute is required.	AnElement TypeB — anAttribute double
Example: <AnElement anAttribute=”14.44e-3”/> Note that all attribute values must be enclosed in quotes, regardless of type.	
This diagram depicts an element with two attributes. The attribute anAttribute is required. The “?” in the circle indicates that the second attribute, anOptionalAttribute, is optional. Both attributes are of type string.	AnotherElement TypeA — anAttribute string ? anOptionalAttribute string
Examples: <AnotherElement anAttribute=”red” anOptionalAttribute=”a string” /> <AnotherElement anAttribute=”blue” />	
The element OneToManyOrElements is the parent of an unordered list of one or more instances of the elements AnElement and AnothertElement. The “+” in the circle indicates the occurrence is one to many and the angled lines indicate this is a choice relationship (“|”) between the children elements.	OneToManyOrElements + AnElement TypeB / AnotherElement TypeA
< OneToManyOrParentElement>...	



The absence of an occurrence bubble declares that one and only one occurrence are allowed. The AnOrParentElement can have one of AnElement or AnotherElement as a child element.	
The '*' in the circle indicates the cardinality (choice) is from 0 to many.	
This diagram depicts an element, From2to3Elements. The element has no type and no attributes. It can have from 2 to 3 sub-elements of either AnElement or AnotherElement.	
This diagram depicts an element, AParentElement, of type AParentElementType. This element has one attribute, attributeOfParent, which is optional. The lines with square corners indicate that occurrences of AnElement and AnotherElement must appear in the order by the illustration on the right where the top element is addressed first and AnotherElement is addressed secondly.	
This diagram depicts a type, AParentElementType, that contains a sequence starting with one of AThirdElement or AFourth element followed by 0-n AnElement and an optional final AnotherElement.	

3.3 Rules concerning the use of XML and XML Schema

The rules required to define syntax and semantics of the 25XX file format notation have been simplified by the adoption of the W3C standards for XML Schema and XML file formats. These two standards are well specified by the W3C. The popularity of these standards has lead to the development of many commercial and open source software tools and libraries that conform to the W3C standards.

A 25XX file begins with the <25XX revision = "2.0"> tag and end with the </25XX> tag. The content between these tags must match the .xsd definition of the 25XX element as defined by the specific IPC-25XX Schema.

3.3.1 File readability and uniformity

A valid 25XX file must conform to the W3C Canonical XML format. The format is defined by the http://www.w3.org/TR/xml-c14n specification. Software tools exist that will take malformed XML and automatically generate Canonical XML (XMLspy, XML authority etc.).

3.3.2 File markers

An optional checksum can be appended following the </25XX> tag. The checksum is an MD5 message digest algorithm (see Internet RFC 1321: http://www.ietf.org/rfc/rfc1321.txt) that is base64 encoded. The checksum starts with the "<" character of the <25XX> tag and ending with the ">" character of the closing </25XX> tag. The checksum follows immediately after the ">" character of the closing </25XX> tag.

3.3.3 File extension

The file extension for IPC-25XX series standards varies according to the domain of the standard.

IPC-2510 .gcm

IPC-2530 .srf

IPC-2570 .pdx

IPC-2580 .cvg

3.3.4 File remarks

The 25XX formats permit file remarks using the standard XML commenting notation. They are only to be used to support debugging software. A parser may ignore and discard remarks when reading a 25XX file. File remarks are never to be used to represent design, manufacturing or equipment messaging information.

3.3.5 Character set definition

The XML standard uses the Unicode character set. This character set covers the characters used in hundreds of written languages. The XML standard allows several of the Unicode encoding formats to be used in an XML file. IPC-25XX standards require the use of the UTF-8 character encoding of the Unicode character set. Although comments and user assigned names may be in any language of choice, all qualified names or enumerated string names **shall** be in English only.

3.4 Data organization and identification rules

The 25XX standards may use a namespace mechanism for XML instance files that is similar to the XML namespace mechanism that was created for managing XML meta-data namespaces. The instance file namespace mechanism prevents collisions between the names used by the different products within a single file. This partitioning of namespaces is necessary because any of the 25XX file may contain information describing an arbitrary collection of products. (Boards, assemblies, panels or components that are products allowed in an IPC-25XX file.) For example, a file could contain descriptions for building multiple electronic assemblies that are manufactured on separate panels. This mechanism also prepares the way for a distributed database of 25XX design data in which the data can be trusted to be universally unambiguous.

3.4.1 Naming elements within a 25XX file

There are two types of names used to name top-level objects (element instances) in a 25XX file. The first type of name is a qualifiedName type. This type includes a prefix in the name that corresponds to a namespace within the 25XX file. The prefix and the globally unique identity of the Namespace are declared in the Namespace element. The second type of name is a shortName type. This type is required to be unique within the 25XX file. The syntax restrictions on short names and qualified names assure that all names will be unique as top-level names within 25XX file.

3.4.2 The use of XML elements and types

The W3C provides a set of general purpose element and attribute types, such as xsd:string, xsd:double, and xsd:datetime. The 25XX format uses these standard types, however the documentation of any 25XX standard has been defined without the use of a namespace prefix for element names within a 25XX file.

Each of the schema elements has a prefix, "`xsd:`", which is associated with the XML Schema namespace through the declaration, `xmlns:xsd="`http://www.w3.org/2000/08/XMLSchema", that appears in the schema element. The prefix `xsd:` is used by convention to denote the XML Schema namespace, although any prefix can be used. The same prefix, and hence the same association, also appears on the names of built-in simple types, e.g. `xsd:`string. The purpose of the association is to identify the elements and simple types as belonging to the vocabulary of the XML Schema language rather than the vocabulary of the schema author.

In XML Schema, there is a basic difference between complex types that allow elements in their content and may carry attributes, and simple types that cannot have element content and cannot carry attributes. There is also a major distinction between definitions that create new types (both simple and complex), and declarations that enable elements and attributes with specific names and types (both simple and complex) to appear in document instances.

New complex types are defined using the `complexType` element and such definitions typically contain a set of element declarations, element references, and attribute declarations. The declarations are not themselves types, but rather an association between a name and constraints that govern the appearance of that name in documents governed by the associated schema. Elements are declared using the "element," and attributes are declared using the "attribute."

3.4.3 Attribute base types (governing templates)

The attribute basic types (SimpleTypes) provided by XML Schema are defined by the W3C. They are easy to distinguish from the IPC-25XX types because the W3C type is always prefixed with "xsd:". The W3C datatypes are defined in http://www.w3.org/2000/10/XMLSchema (XML Schema Part 2).

Table 2 defines those W3C basic types that are used to define attributes in the 25XX schema. The xsd:string type is constrained to create specific base types for special purpose strings, such as `qualifiedName` and `shortName`. The rules for special number types and the date format are also defined. Table 3 defines those basic types that have been standardized for use within the IPC-25XX format.

Table 2 Governing template basic types defined by W3C

xsd:string	A W3C standard data type for a Unicode character string. The characters are from the UTF-8 character set as defined in http://www.ietf.org/rfc/rfc2279.txt.
xsd:double	A W3C standard data type for a binary floating-point number. The W3C definition of xsd:double is in http://www.w3.org/TR/xmlschema-2/. The xsd:double is a number where the value can be positive, negative, integer or floating point, with at least 7 digits of precision. Numbers are assumed to be positive but can be explicitly designated as positive by preceding the number with a '+' (ASCII decimal 43) character. Negative numbers must be explicitly designated as negative by a preceding '–' (ASCII decimal 45) character. An internal representation of an IEEE double precision floating-point number is assumed. This range of values for IEEE doubles is defined as 3.4x10-38 ≤ value ≤ 3.4x10+38. The format for representing a double is the same as the format used in the computer languages C, Perl, Python, or TCL. For example, all the following are legal numbers: 1.005 ; 0.01; .01; -2.334e-33; .224e-2
xsd:nonNegativeInteger	A W3C standard data type for non-negative integer numbers. The W3C definition of xsd:nonNegativeInteger is in http://www.w3.org/TR/xmlschema-2/. The range of values allowed are 0 ≤ value ≤ 2147483647 (the non-negative values that fit in a 32 bit signed integer).
xsd:positiveInteger	A W3C standard data type for positive integer numbers. The W3C definition of xsd:positiveInteger is in http://www.w3.org/TR/xmlschema-2/. The range of values allowed are 1 ≤ value ≤ 2147483647 (the positive values that fit in a 32 bit signed integer).
xsd:dateTime	The W3C standard data type for the current date and time is xsd:dateTime. (See http://www.w3.org/TR/NOTE-datetime-970915.html.) The following formats from the W3C specification are recommended for 258X files: Complete date plus hours, minutes and seconds: YYYY-MM-DDThh:mm:ssTZD (e.g. 1997-07-16T19:20:30.4536+01:00) Complete date plus hours, minutes, seconds and a decimal fraction of a Second: YYYY-MM-DDThh:mm:ss.sTZD (e.g. 1997-07-16T19:20:30.45+01:00) where: YYYY = four-digit year MM = two-digit month (01=January, etc.) DD = two-digit day of month (01 through 31) Hh = two digits of hour (00 through 23) (am/pm NOT allowed) Mm = two digits of minute (00 through 59) Ss = two digits of second (00 through 59) S = one or more digits representing a decimal fraction of a second TZD = time zone designator (Z or +hh:mm or –hh:mm)
xsd:anyURI	A W3C standard data type for hyperlinks. The W3C definition of xsd:anyURI is in http://www.w3.org/TR/xmlschema-2/.

xsd:unsignedByte	The W3C standard for an unsigned byte (an unsigned 8 bit integer with a value between 0-255.) The W3C definition of xsd:unsignedByte is in http://www.w3.org/TR/xmlschema-2/.
xsd:base64Binary	The data is encoded using base64. (see IETF *RFC 1421* for the base64 algorithm and http://www.w3.org/TR/xmlschema-2/#base64Binary)

Table 3 Governing template basic types defined by IPC

qualifiedName	The `qualifiedName` data type is a data type defined for the 258X series. The type is a restricted xsd:string data type where the pattern of the string must match the regular expression “[a-zA-Z][a-zA-Z0-9_-]*:.+”. The definition of the qualifiedName data type is: <xsd:simpleType name = “qualifiedName”> <xsd:restriction base = “xsd:string”> <xsd:pattern value = “[a-zA-Z][a-zA-Z0-9_-]*:.+”/> </xsd:restriction> </xsd:simpleType> An example of a string that matches the pattern is: “prefix:name”. The “prefix” is a `Namespace` name. The “name” is the name of an object within the Namespace.
nonNegativeDouble	The `nonNegativeDouble` data type is defined for the 25XX series. The type restricts an xsd:double to positive numbers, inclusive of 0. The non-negative range of values for IEEE doubles is defined as 0.0 ≤ value ≤ 3. $4x10^{38}$.
Xpath	The xpath data type is a data type defined for the 25XX series. The type is a restricted xsd:string data type where the pattern of the string must be a legal Xpath as defined in W3C http://www.w3.org/TR/xpath.
shortName	The `shortName` data type is a data type defined for the 25XX series. The type is a restricted xsd:string data type where the pattern of the string must match the regular expression “[a-zA-Z][a-zA-Z0-9_-]*”. The xsd definition of the shortName data type is: <xsd:simpleType name = “shortName”> <xsd:restriction base = “xsd:string”> <xsd:pattern value = “[a-zA-Z][a-zA-Z0-9_-]*”/> </xsd:restriction> </xsd:simpleType> An example of a string that matches the pattern is “bob_24”
mimeType	The `mimeType` data type is a restricted xsd:string type that matches IETF MIME type definitions. (e.g. text/html, application/postscript).

3.4.3.1 Qualified name convention

The IPC-25XX file supports two types of qualified names. One is a basic qualifiedName; the second is a complete qualifiedName as shown in Table 3.

A basic qualifiedName is composed of at least one letter, followed by any number of letters, numbers, underscores, or hyphens. To form a complete qualifiedName, one can optionally prefix a basic qualifiedName with a colon delimited path, where each

step along the path is constructed the same way as the basic qualified name. This permits sorting of sort names into a hierarchy (see Table 3).

Examples of basic qualified names are:

“KarenSingleBoard”
“MultilayerStrategy”
“StandardPrimitiveShapes”

Examples of complete qualified names are:

“Set1:KarenSingleBoard”
“Set1:MultilayerStrategy”
“Set1:StandardPrimitiveShapes”

3.4.4 Coordinate system

Any geometry defined in any 25X file is defined in a Cartesian coordinate system. The x coordinates become more positive going from left to right (west to east). The y coordinates become more positive going from bottom to top (south to north). The primary side (TOP) of the board, coupon, or panel is in the x-y plane of the coordinate system with the primary side facing up.

4 Message representation

4.1 Naming conventions

Naming conventions of this document are identified as examples and are located between < >.

4.1.1 XML Element names

XML element names are written with normal text with small letters, but starting with capital. In case of multi word name, each word is started with capital letter and words are written together.

Example: Element1, MultiWordElement2

4.1.2 XML Attributes

Attribute names are written with normal text with small letters and starting also with small letter. In case of multi word name, each word is started with capital letter and words are written together.

Example: attribute1, multiWordAttribute2

4.1.3 Enumerations

Enumerations are marked in the attribute type column as ‘String (enumerated)’

Enumerations are written all in capital letters. If an enumerated string contains a word and a number they are written together; if they are multiple words they shall be separated by an underscore “_”.

Example: ENUMERATION1, MULTI_WORD_ENUMERATION2

Meanings of each enumeration can be defined early in the documentation prior any attribute table. This should be the case if enumeration occurs only in one place of the document.

If enumeration is more generic and is used same way in 2 or more locations, enumeration are opened in tables located in the clause where they occur.

4.2 Generic structure of Message / Element

4.2.1 ElementNameHere

Description: This place is written the description of the element or message

AttributeEnum:

ENUM1 | ENUM2

ENUM1: Definition of the meaning of enumeration ENUM1.

ENUM2: Definition of the meaning of enumeration ENUM2.

Attribute / Element Name	Attribute / Element Type	Description	Cardinality
ElementName	ElementType	Generic structure descriptions	1-1
attributeEnum	string (enumerated)	Describtion of the attribute ENUM1 | ENUM2	1-1
attribute2	Double	Describtion of the attribute.	0-1
attributeOpt1	String	Describtion of the attribute.	1-1
Element1	See '#Chapter'	Describtion of the element.	1-1
Element2	See '#Chapter'	Describtion of the element.	1-n
ElementOpt1	See '#Chapter'	Describtion of the element.	0-1
ElementOpt2	See '#Chapter'	Describtion of the element.	0-3
ElementChoice1.1	See '#Chapter'	Describtion of the element.	1-1
ElementChoice1.2	See '#Chapter'	Describtion of the element.	1-1

```
<ElementNameHere
      attributeEnum="ENUM1"
      attribute2="0.5"
      attributeOpt1="any_string">
            <Element1/>
            <Element2/>
            <Element2/>
            <ElementOpt2/>
            <ElementChoice1.2/>
</ElementNameHere>
```

An attribute table presents the “flat” representation of the titled element. Presentation of any internal elements data is not allowed, but rather must be presented as a separate element definition. If an element is internal and associated only with this element, it may be presented as subchapter of this element. If the defined sub-element is used in 2 or more locations, it is recommended that this sub-element be presented in the chapter ‘Nested elements’

Table organization order:

1. Mandatory elements
2. Mandatory attributes
3. Optional attributes
4. Optional elements

For association reasons order of elements and attributes may be organized and associated attributes/elements can be grouped together.

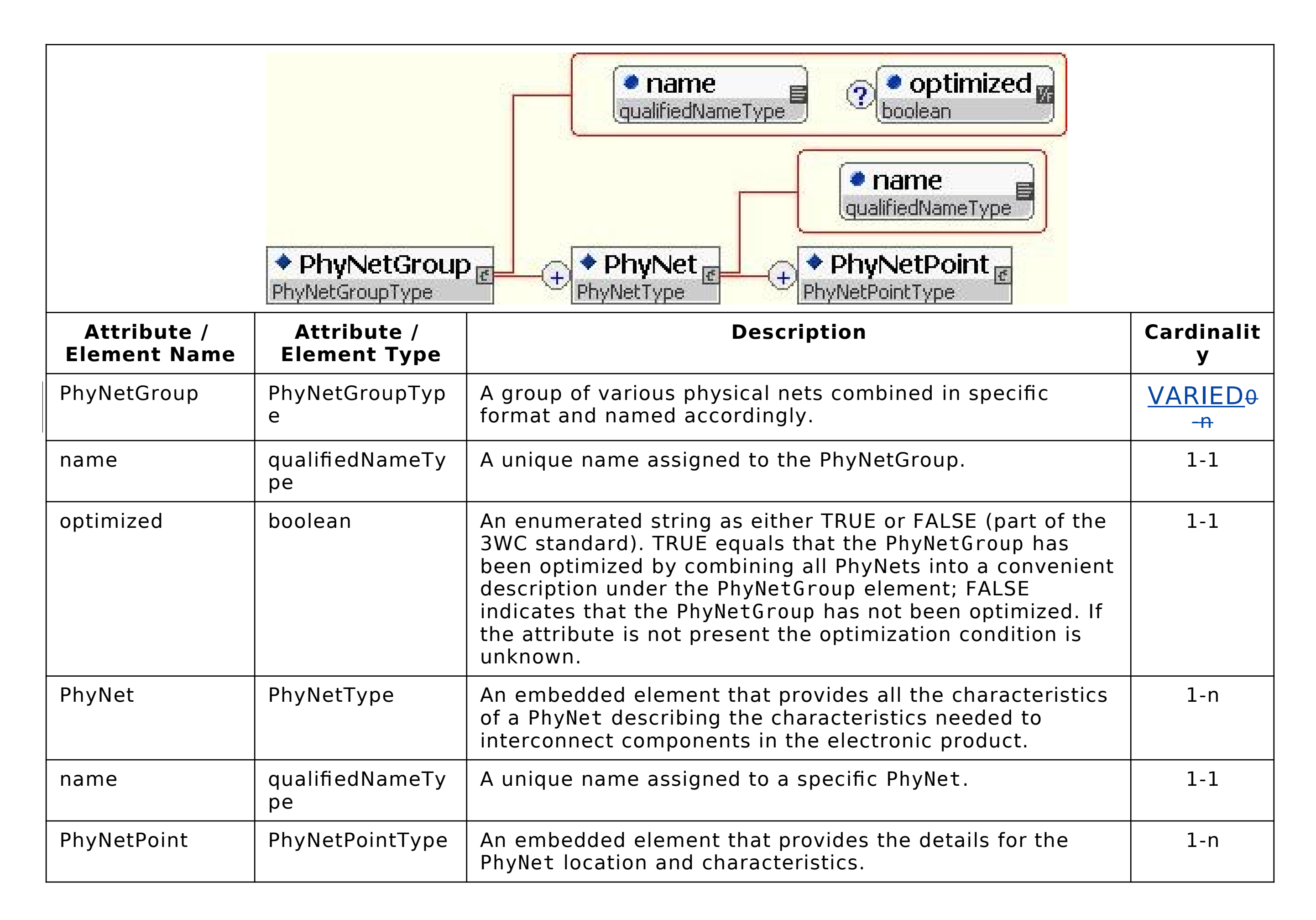
The same order as in table **must** be followed in XML Schema definition. This comes from the use of XML Schema Sequence –structure as the default in the message elements.

Cardinality field
Accepted values and structure of the field is
{0…N} – {1…N, n}, where
N is any total number greater than 0 and
n is any number from group N.

If XML Schema Choice –structure is used in the element then it must be presented as "optional" unless these are of a "substitution group" nature.

4.3 Graphic and table combination

The order in which attributes or elements occur within a table can be random per the domain; they do not need to be in alphabetical order. The order should be based on the domain conditions of the element family (parent and children). When an element consists of several attributes, the attribute should be listed with the mandatory attributes first followed by the optional attributes. In addition, each table should start with the parent element description, even though that element may already have been defined as a child of a previous XML schema hierarchy. The following is an example of the graphic included as a part of the table.



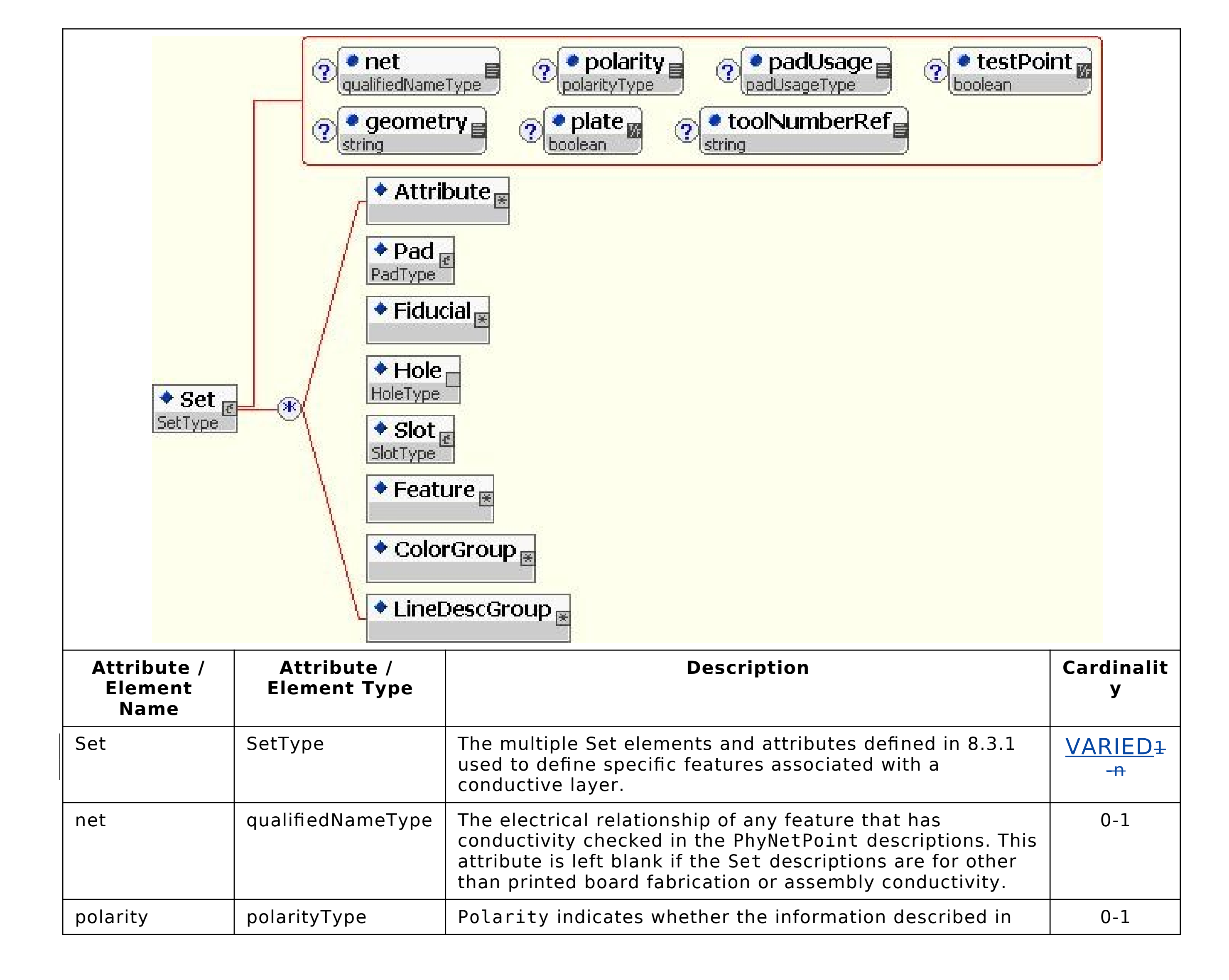
Attribute / Element Name	Attribute / Element Type	Description	Cardinality
PhyNetGroup	PhyNetGroupType	A group of various physical nets combined in specific format and named accordingly.	VARIED~~0-n~~
name	qualifiedNameType	A unique name assigned to the PhyNetGroup.	1-1
optimized	boolean	An enumerated string as either TRUE or FALSE (part of the 3WC standard). TRUE equals that the PhyNetGroup has been optimized by combining all PhyNets into a convenient description under the PhyNetGroup element; FALSE indicates that the PhyNetGroup has not been optimized. If the attribute is not present the optimization condition is unknown.	1-1
PhyNet	PhyNetType	An embedded element that provides all the characteristics of a PhyNet describing the characteristics needed to interconnect components in the electronic product.	1-n
name	qualifiedNameType	A unique name assigned to a specific PhyNet.	1-1
PhyNetPoint	PhyNetPointType	An embedded element that provides the details for the PhyNet location and characteristics.	1-n

As can be seen from the example above, the first line item in the table is the element "PhyNetGroup." This is followed by the attributes with the mandatory attribute "name" being first. This would then be followed by children elements that are also related to the parent PhyNetGroup. The mandatory element PhyNet is next, followed

by its mandatory attribute and finally an indication of the child element "PhyNetPoint."

Since many of the standards require an example of an instance file using the schema represented by the graphic or the table, this would be contained in a boxed in area below the description of the elements and attributes. It should be relatively clear in the documentation that the graphic, the table descriptions, and the instance description example all relate to the subject of the paragraph heading. Although this example is intended to provide the most robust explanation, the table represents a mandatory requirement where the graphic and the instance of that schema are optional.

When an element provides a choice of different sub-elements, it can usually be identified clearly in a graphic and so it is not necessary to split the table in order to show that choice. However, the table must indicate all items of the choice and the cardinality would enumerate their characteristics. This is shown in the following example:



Attribute / Element Name	Attribute / Element Type	Description	Cardinality
Set	SetType	The multiple Set elements and attributes defined in 8.3.1 used to define specific features associated with a conductive layer.	VARIED ~~1-n~~
net	qualifiedNameType	The electrical relationship of any feature that has conductivity checked in the PhyNetPoint descriptions. This attribute is left blank if the Set descriptions are for other than printed board fabrication or assembly conductivity.	0-1
polarity	polarityType	Polarity indicates whether the information described in	0-1

		the Set is POSITIVE | NEGATIVE. A NEGATIVE connotation can be used to describe the removal of a dark field to the specific dimensions described for another attribute. Thus, a surface that contains islands may have the islands described in a negative format.	
padUsage	padUsageType	An indication as to the usage of any pad that becomes a part of the LayerFeature Set. The descriptions are enumerated strings and must be one of the following: TOE | VIA | GLOBAL_FIDUCIAL | LOCAL_FIDUCIAL | TOOLING_HOLE | NONE.	0-1
testPoint	boolean	An enumerated string as either TRUE or FALSE (part of the 3WC standard). TRUE indicates that the feature is a candidate for a test point used for either in-circuit or functional testing. FALSE indicates that it is not.	0-1
geometry	string	An identification to describe the overall geometry of the features contained in the Set and their particular application to the electronic product.	0-1
plate	boolean	An enumerated string as either TRUE or FALSE (part of the 3WC standard). TRUE indicates that the feature is plated in a secondary operation . FALSE indicates that it is not.	0-1
toolNumberRef	string	A reference to the tool number defined in the DrillTool instance of the Layer section. This feature is used to associate the tool with features that are part of the Set.	0-1
Attribute	ABSTRACT	A substitution group that may be any of a group of enumerated string descriptions or a unique string for a condition not addressed by the standard attributes. The Attribute is associated with the LayerFeature Set.	0-n
Pad	PadType	A series of pads that are associated with the LayerFeature Set.	0-n
Fiducial	ABSTRACT	A substitution that consists of four elements that may be used to replace the fiducial element. When the Fiducial element is substituted it **shall** be by a Global, Local, BadBoardMark or GoodPanelMark.	0-n
Hole	HoleType	A series of holes associated with the LayerFeature Set.	0-n
Slot	SlotType	A series of slots associated with the LayerFeature Set.	0-n
Feature	ABSTRACT	A substitution group of any predefined StandardShape or UserShape that may be instantiated by the user at the time the feature contained in the set is described.	0-n
ColorGroup	ABSTRACT	A substitution group that permits assigning a particular color through instantiating the three basic colors or by providing a reference to a predefined Color in DictionaryColor.	0-n
LineDescGroup	ABSTRACT	A substitution group that specifies the LineWidth and LineEnd characteristics of a Feature that requires that description. If a predefined feature is instantiated the presents of a LineDescGroup will override the previously defined LineDesc.	0-n

The example shows that there are several choices that can be instanced as a part of the Set element. Each of these may be 1 or more or the children element characteristics.

4.4 Substitution Groups

The IPC-25XX standards may also use the concept of substitution within the XML schema. Various groups of elements are identified in the body of the standard and have been designated as having a specific focus or purpose. Within any schema,

these substitution groups are provided with a name. When a group exists and if they are required according to the instances of the schema, it is mandatory that the substitution name be replaced by one of the acceptable descriptions identified within the group.

Often a schema needs to specify that one of several different XML Elements can be used with equal validity. As an example, in every case where a Triangle can be used, it is also permissible to use a Diamond, Hexagon, Octagon, Oval, or one of several others: even though these shapes are quite different, they are equivalent as far as the schema is concerned.

A substitution group consists of two types of elements: a “head”, and elements which may substitute for the head. Furthermore, when the head is denoted as ABSTRACT, the substitution is required, rather than optional. The IPC-25XX, should identify the heads of all substitution groups are ABSTRACT. Thus, it means that a valid instance document is not allowed to contain a StandardPrimitive element, but instead, (where StandardPrimitive is called for in the schema) a Triangle, Diamond, Hexagon, etc. must be used.

It should be noted that the head of one substitution group may be used within a different substitution group. As an example, the StandardPrimitive element is part of the StandardShape substitution group, which in turn is part of the Feature substitution group. This means that a Triangle, Diamond, Hexagon, etc may be used wherever a Feature or StandardShape is called for, as well as wherever a StandardPrimitive is called for.

There are 13 substitution groups in the following example. These are shown in the following table.

SubstitutionGroups
- Attribute
- ColorGroup
- Feature
- Fiducial
- FirmwareGroup
- FontDef
- LineDescGroup
- PolyStep
- Simple
- StandardPrimitive
- StandardShape
- UserPrimitive
- UserShape

Attribute / Element Name	**Attribute / Element Type**	**Description**	**Number of substitution elements** ~~**Cardinality**~~
Attribute	ABSTRACT	A substitution group that permits the substitution of the Attribute element when it is a child of the parent Component, LogicalNet, Set, or Step elements.	4
ColorGroup	ABSTRACT	A substitution group that permits the substitution of the Color element when it is a child of the parent FinishType, Set, or Text Elements.	3

Feature	ABSTRACT	A substitution group that permits the substitution of the Feature element when it is a child of the parent Set element.	1
Fiducial	ABSTRACT	A substitution group that permits the substitution of the Fiducial element when it is a child of the parent Set element.	1
FirmwareGroup	ABSTRACT	A substitution group that permits the substitution of the FirmwareGroup element when it is a child of the parent Firmware element.	1
FontDef	ABSTRACT	A substitution group that permits the substitution of the FontDef element when it is a child of the parent EntryFont element.	1
LineDescGroup	ABSTRACT	A substitution group that permits the substitution of the LineDescGroup element when it is a child of the parent Outline, Polyline, or Set elements.	3
PolyStep	ABSTRACT	A substitution group that permits the substitution of the PolyStep element when it is a child of the parent Polyline or Polygon elements.	2
Simple	ABSTRACT	A substitution group that permits the substitution of the Simple element when it is a child of the parent DfxMeasurement, Glyph, or Slot elements.	3
StandardPrimitive	ABSTRACT	A substitution group that permits the substitution of the StandardPrimitive element when it is a child of the parent EntryStandard element.	1
StandardShape	ABSTRACT	A substitution group that permits the substitution of the StandardShape element when it is a child of the parent LayerPad or Pad elements.	2
UserPrimitive	ABSTRACT	A substitution group that permits the substitution of the UserPrimitive element when it is a child of the parent EntryUser element.	1
UserShape	ABSTRACT	A substitution group that permits the substitution or classification of a higher level substitution group. The UserShape element may be used to further classify Feature. In so doing, UserShape can be substituted by a UserPrimitive or UserPrimitiveRef.	0

4.5 Example message

4.5.1 Element: MaterialHandler

Description: Material handler (e.g. tape, bowl or tray feeder, screw shooter) is used to supply items to assembly operation. The attributes associated with a feeder would be used if the component is located on a feeder and the attributes associated with a tray would be used if the component is located on a tray feeder.

Attribute / Element Name	Attribute / Element Type	Description	Cardinality
materialSupplyArea	string	Unique area of material (i.e. component) supplies found on the equipment	1-1
FeederType	string	A specific type of a feeder	1-1
FeederId	string	Identifier of the feeder	0-1
feederDivision	string	Unique location within a feeder or a tray	1-1
Feeder	string	Detailed information about feeder	1-1
TrayFeeder	string	Detailed information about tray feeder	1-1

4.5.2 Sub Element: Feeder

Description: The sub-element “Feeder” is used to correlate the characteristics of the feeder mechanism with that of the material handler and the track identification.

Attribute / Element Name	Attribute / Element Type	Description	Cardinality
TrackId	1..n	Unique location on the equipment. Sometimes referred to as slot.	1-1
materialHandlerTableId	String	Identifier for the specific materialHandler table which contains bank of feeders	0-1

4.5.3 Sub Element: TrayFeeder

Description: The sub-element “TrayFeeder” is used to correlate the characteristics of the tray and its feeder mechanism with that of the material handler when material is provided in a tray.

Attribute / Element Name	Attribute / Element Type	Description	Cardinality
trayFeederTower	1..n	Tray feeder tower number	1-1
TraySection	String	Tray section	1-1
trayFeederLocation	1..n	Tray feeder location number	0-1
TrayId	String	Identifier of tray	0-1

```
<MaterialHandler
      materialSupplyArea="2"
      feederType="8mm Tape"
      feederId="65432-1"
      feederDivision="1">
      <Feeder
            trackId="1"
            materialHandlerTableId="1"/>
</MaterialHandler>

<MaterialHandler
      materialSupplyArea="1"
      feederType="TrayMagazine"
      feederId="C453-4321"
      feederDivision="2">
      <TrayFeeder
            trayFeederTower="2"
            traySection="4"
            trayFeederLocation="3"
            trayId="4342"/>
</MaterialHandler>
```

The example shows a separate description of 3 elements (MaterialHandler, Feeder, TrayFeeder). It is an option to separate the different elements within the IPC-25XX standard. It is also possible that one graphic could describe the parent element with its two children as shown in the previous example. However, there will be times when breaking the schema into the sub-elements is important for the comprehension of the requirements.

5 Units enumerations in the IPC-25XX standards

Units are used in various places in the standards. Units will follow SI units naming conventions as open written form. The following enumerations **shall** be used.

The main differences are:

- Follows the Enumerations naming conventions
- Mathematical symbols are used for:
  - * for multiplication (UTF-8 character number 0042 = 002Ah)
  - / for deviation (UTF-8 character number 0047 = 002Fh)

- ^ for exponent (UTF-8 character number 0094 = 005Eh)
- Exponent and numbers are used for expression of power. Like square (^2), cubic (^3).

For more information about SI-units:

http://physics.nist.gov/cuu/

http://physics.nist.gov/cuu/Units/units.html

For more information about Unicode and UTF characters:

http://www.unicode.org/charts/ → UTF-8 = Basic Latin + Latin-1 Supplement

5.1 List of units

If an application uses a quantity or unit that is not listed in the following chapters then the application must document that fact clearly. Any added name or unit should follow the International System of Units (SI-units). But if a quantity is listed (like length) all allowed units are presented in the following table.

Primary indicator is used for primary unit (i.e. SI-unit) for a quantity. The primary unit should be used whenever possible.

5.1.1 Time

SECOND	Primary

It should be noted that this applies only to those locations where time is used as the unit within a measurement There are some special definitions for time that are used directly in message fields. Many messages can directly use the W3C attribute definitions of time as defined in:

dateTime (http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/#dateTime) and

duration (http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/#duration)

5.1.2 Temperature

CELSIUS	
FAHRENHEIT	
KELVIN	Primary

5.1.3 Length

INCH	
METER	Primary

5.1.4 Angle

DEGREE	
RADIAN	Primary

5.1.5 Area

INCH^2	
METER^2	Primary

5.1.6 Volume

INCH^3	
METER^3	Primary

5.1.7 Mass

KILOGRAM	Primary
OUNCE	

5.1.8 Linear Velocity

INCH/SECOND	
METER/SECOND	Primary

5.1.9 Angular Velocity

RADIAN/SECOND	Primary
REVOLUTION/SECOND	

5.1.10 Linear Acceleration

INCH/SECOND^2	
METER/SECOND^2	Primary

5.1.11 Angular Acceleration

RADIAN/SECOND^2	Primary
REVOLUTION/SECOND^2	

5.1.12 Flow

METER^3/SECOND	Primary
FEET^3/SECOND	

5.1.13 Electrical (collection)

AMPERE	Electric Current
HERZ	Frequency
JOULE	Energy, work, quantity of heat
OHM	Electric resistance
VOLT	Electric potential difference, electromotive force
WATT	Electric power

5.1.14 Mechanical (collection)

NEWTON	Force
NEWTON*METER	Moment of force, Torque
PASCAL	Pressure, stress

5.1.15 Others (collection)

METER/SECOND^3	Jerk. Derivative of acceleration
PERCENT	
RELATIVE_HUMIDITY	
PIXEL	

References

Checksum

The digest provides a 128-bit checksum of the 258X file contents. The MD5 signature must be base64 encoded (see IETF RFC 1421 for the base64 algorithm) to convert the MD5 signature to a US-ASCII, base64 string. An end of line character will indicate the end of the base64 encoded MD5 signature.